MATLAB FOR FINANCE 2015 - HOMEWORK ASSIGNMENT

General remarks:

Submission. Submission date is Saturday, **October 17th**, at **11:00 pm**. Each of the two parts (GARCH, Portfolio selection) has to be implemented and submitted in a separate MATLAB script labled according to the name of the part (for example for part 1: garch_analysis). Put the two MATLAB scripts as well as ANY MATLAB function that you additionally did use - both the ones that you have additionally implemented yourself and the functions you did rely on from the course (hist_stock_data, getPrices,...) - into a git repository hosted on github, and send a link to the repository in an email with subject "MATLAB for Finance: Assignment" to Christian.Groll@stat.uni-muenchen.de.

Grading criteria. Besides the correct implementation of the exercises, you are also required to keep your code as readable and comprehensible as possible. That is, use comments whenever they help to make your code more readable. Furthermore, keep an eye on the performance speed of your code, whenever you get to time-consuming passages. The individual exercises will be weighted equally!
Git requirements: your repository must consist of at least 10 commits made on at least 5 different days!
Note: depending on your version of MATLAB and your operating system, you probably could encounter out-of-memory errors in some of the more demanding code segments. If such an error occurs, simply write the orginally required code as comment in your file, and conduct the simulation with appropriately adjusted iteration number and sample size.

Part 1. GARCH

As already seen in the course, GARCH processes differ from AR-processes in that autoregressive processes are conditional mean models, with today's mean return depending on past observations, while GARCH models entail a conditional variance equation, so that today's volatility is depending on the past. This way, GARCH models are able to capture the time-varying patterns of financial return series with respect to second moments. However, besides the improvement with respect to volatility clusters, GARCH models are also able to introduce fat tails into the unconditional distribution of the model. Using the default model of MATLAB's toolbox, given by

$$Y_t = C + \epsilon_t,$$

where

$$\epsilon_t \sim z_t \sigma_t, \text{ and } z_t \sim \mathcal{N}(0,1),$$

and

$$\sigma_t^2 = \kappa + G_1 \sigma_{t-1}^2 + A_1 \epsilon_{t-1}^2,$$

you shall examine how already this default model is able to reproduce the two stylized facts volatility clusters and fat tails.

1. Volatility Clusters

Exercise 1.

- Load historic stock prices of the company Deutsche Bank into the MATLAB workspace using the function getPrices. Specify January 1st, 2000, as first observation of the sample period, and January 1st, 2015, as last observation.
- Transform stock prices to percentage logarithmic returns.
- Estimate the parameters of the GARCH default model using the already existing built-in function `estimate` for class GARCH from the econometrics toolbox. Hint: you need to specify conditional mean and conditional variance in order to be able to infer the estimated innovations with `infer`.
- Based on the estimated parameter values of the model, conduct a backtesting procedure similar to the one in the slides of the course. That is, for each past day, extract conditional mean and variance values, estimate VaR for a confidence level of 0.95 based on an assumption of normally distributed innovations, and check whether an exceedance has occurred, or not. Calculate the overall exceedance frequency, and visualize past observations, exceedances and VaR estimates in different colors in a figure. Let MATLAB display the frequency of exceedances.
- In order to further compare the dynamics of the process specified by the fitted GARCH default model with the dynamics of real world observations, use the already existing function `simulate` in order to simulate a path of length 40000 from the fitted model. Again, compare the empirical autocorrelation function of squared returns for real world data with the counterpart of the default model.
- Furthermore, take a deeper look at the characteristics of the model by simulating 3 sample paths of length equal to the sample size, and showing a large figure with four subplots: real observed data at the top, simulated sample paths below. For better comparison, set all y-limits to the automatically chosen limits of the real data sample. Without knowledge of the arrangement, would you recognize the real historic path?

2. Fat Tails

In order to get an impression about the unconditional distribution associated with a GARCH model, you now shall compare non-parametrically estimated probability density functions of real data and simulated data.

Exercise 2.

- In order to come up with a non-parametric estimate of the density function for the real world data sample, use function `ksdensity` to show a Kernel smoothing estimate for the underlying data.
- As a benchmark model, first estimate parameters of a normal distribution which is fitted to the real world data sample. Include the associated probability density function in red color into the figure.
- Visualize the unconditional probability density associated with GARCH, by simulating 40000 observations of the fitted default GARCH model and applying a Kernel smoothing density estimate with function `ksdensity`. Include the kernel density estimate into the figure in green color.
- At last, you now shall compare the results to a GARCH model with innovations distributed according to a Student's t-distribution, in contrast to the normal distribution used so far. Therefore, create a GARCH specification object with property "distribution" set to "T". Use the modified GARCH specification structure as additional input into `estimate` in order to fit a GARCH model with t-distributed innovations to the real world data sample.
- Simulate a path of length 40000 of the new GARCH model, and visualize its unconditional probability density estimate by application of `ksdensity` and inclusion of the estimate into the figure in black color.
- Which model does best replicate the leptokurtosis of the real world data sample?

Part 2. Portfolio selection

In 1952 The Journal of Finance published a paper of Markowitz, titled "Portfolio Selection". In this paper, Markowitz examined diversification effects in portfolio management. Thereby, investors are assumed to maximize their expected portfolio return, conditional on a given level of portfolio risk, which is measured in standard deviations. Hence, for a given universe of assets X_i, each with discrete percentage return denoted by r_i and estimated expected return μ_i and standard deviation σ_i, investors carefully choose the portfolio proportions in order to maximize their expected portfolio return.
Since portfolio returns are given as a sum of weighted individual returns in the discrete case, the formulas for portfolio expectation and portfolio variance are linear and given by

$$\begin{aligned}\mu_P &= \mathbb{E}\left[\sum_{i=1}^{N} w_i r_i\right] \\ &= \sum_{i=1}^{N} w_i \mathbb{E}\left[r_i\right] = \sum_{i=1}^{N} w_i \mu_i\end{aligned}$$

and

$$\begin{aligned}\sigma_P^2 &= \mathbb{E}\left[\left(\sum_{i=1}^{N} w_i r_i - \mu_P\right)^2\right] \\ &= \mathbb{E}\left[\left(\sum_{i=1}^{N} w_i \left(r_i - \mu_i\right)\right)^2\right] \\ &= \sum_{i=1}^{N} w_i^2 \sigma_i^2 + \sum_{i=1}^{N} \sum_{j \neq i} w_i w_j \sigma_{ij},\end{aligned}$$

with covariance σ_{ij}.

Exercise 3.

- Load data of all 30 components of the German stock index DAX into the MATLAB workspace using the function getPrices. Specify January 1st, 2000, as first observation of the sample period, and January 1st, 2015, as last observation.
- For each component, use the maximum size of observations available in order to estimate expected **discrete** percentage return μ_i and standard deviation σ_i. Note: you cannot use function price2retWithHolidays as it computes **logarithmic** returns!
- For each pair of stocks, use the maximum size of coinciding return observations in order to estimate the correlation coefficient ρ_{ij}.
- Plot a histogram with 20 bins of all $30 \cdot 29/2$ estimated correlation coefficients.
- Let MATLAB display the ticker symbols associated with the pair of stocks with the highest estimated correlation coefficient.
- Create a figure with estimated expected returns on the y-axis, and estimated standard deviations on the x-axis, and include for each stock one point at coordinates (σ_i, μ_i). The points shall be plotted in red color.
- Simulate 200 positive random portfolio weights for the stocks of DBK.DE and DPW.DE, denoted by X_{DBK} and X_{DPW}. Keep in mind, that portfolio weights have to sum up to

1. For each pair of weights $\left\{w_{DBK}^k, w_{DPW}^k\right\}$, $k = 1, ..., 200$, determine expected portfolio return and standard deviation $\left(\sigma_P^k, \mu_P^k\right)$, and include a blue mark at the respective point in the coordinate system. Also, highlight the points of both individual portfolio components $(\sigma_{DBK}, \mu_{DBK})$ and $(\sigma_{DPW}, \mu_{DPW})$ with slightly larger points in green color.

- Create a new figure, again with each (σ_i, μ_i) pair included in red. Conduct the same analysis for a portfolio consisting of four assets, namely DBK.DE, DPW.DE, TKA.DE and CBK.DE. This time, create 50000 random portfolio weights vectors $w = (w_1, \ldots, w_4)$, with constraints $w_i > 0$ for $i = 1, ..., 4$ and

$$\sum_{i=1}^{4} w_i = 1.$$

  Try to generate random weights in a way that on average no asset will be taken with lower weights than the rest. Hint: try to linearize the calculation of the portfolio standard deviation in order to improve the performance of your code.
- Analyze your findings: which points in your figure are desireable outcomes, and which properties of the set of optimal points do you expect from an analytical solution of the problem, based on your graphical findings?